SELF DRIVING CAR

A SYNOPSIS

Submitted by

VINAYAK LAL
UDIT TIWARI
ASHWINI JOSHI
MEENAL SRIVASTAVA

Bachelor of Technology

IN

Computer Science and Information Technology

TO

School of Computer Science and Information Technology

SYMBIOSIS UNIVERSITY OF APPLIED SCIENCES, INDORE

Under the guidance of

DR. NEHA GUPTA

MAY 2021

TABLES OF CONTENTS

S.N.	Content	Page No.
1.	*Title*	*2*
2.	*Problem Definition*	*2*
3.	*Objective*	*2*
4.	*Scope*	*3*
5.	*Literature Reviews / Feasibility Study*	*3*
6.	*Proposed Methodology*	*5*
7.	*Facilities required*	*8*
8.	*Timeline*	*9*
9.	*Team*	*10*
	Bibliography	*10*

TITLE

Self-Driving car

PROBLEM DEFINATION

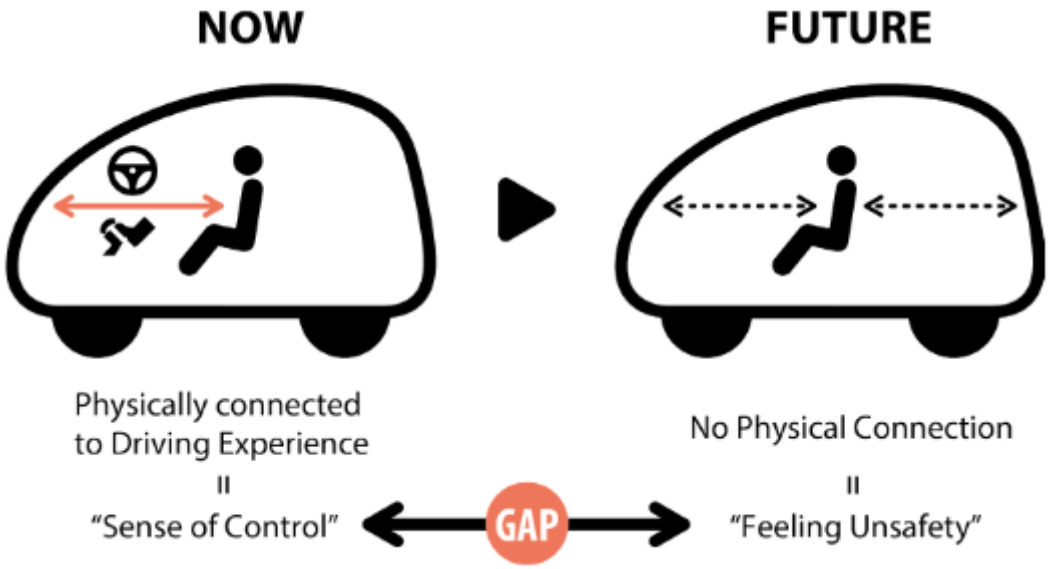
1.25 million people die in road accidents globally every year, or on an average 3287 deaths per day and 20-50 million people get injured and disabled for life annually, as reported by World Health Organization (WHO). International Road Federation (IRF), Geneva report says that India has the highest number of road deaths and ranking first while accounting for 10% of global road accidents. It is found that 78% of road accidents are due to drivers' fault or human error in driving and it is interesting to address this human issue with deployment of technology in the form of Driverless Car. This helps in significant savings in time and will reduce the number of road accidents.

OBJECTIVE

1. Reduce traffic congestion (30% fewer vehicles on the road)
2. Cut transportation costs by 40% (in terms of vehicles, fuel, and infrastructure)
3. Improve walkability and liveability
4. Free up parking lots for other uses (schools, parks, community centres).
5. Reduce urban CO 2 emissions by 80% worldwide as they work on electric batteries.

SCOPE

1. Electric and Autonomous Bus Service that can transport up to 15 passengers safely, suitable for Urban Areas, Industrial Areas, Airports or Amusement Parks.
2. Google Self Driving Car designed without a steering wheel and pedals.
3. Tesla Autopilot which uses RADAR Technology allowing for detection of both hard and soft objects.



LITERATURE REVIEW / FEASIBILITY STUDY

Greater Road Safety

Automation can help reduce the number of crashes on our roads.

Government data identifies driver behaviour or error as a factor in 94 percent of crashes, and self-driving vehicles can help reduce driver error.

Higher levels of autonomy have the potential to reduce risky and dangerous driver behaviours. The greatest promise may be reducing the devastation of impaired driving, drugged driving, unbelted vehicle occupants, speeding and distraction.

Greater Independence

Full automation offers more personal freedom.

People with disabilities, like the blind, are capable of self-sufficiency, and highly automated vehicles can help them live the life they want. These vehicles can also enhance independence for seniors.

Ride-sharing of HAVs could reduce costs of personal transportation, providing more affordable mobility.

More Productivity

Wide-scale deployment of HAVs could allow drivers to recapture time.

In the future, HAVs could offer the convenience of dropping vehicle occupants at their destination, whether an airport or shopping mall, while the vehicle parks itself.

Reduced Congestion

Several causes of traffic congestion could be addressed by HAVs.

Fewer crashes or fender benders mean fewer roadway backups.

Environmental gain

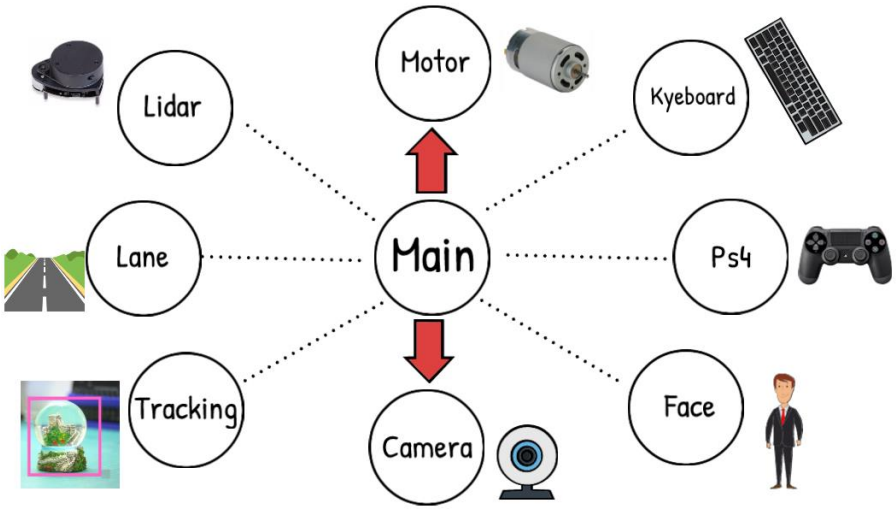
HAVs have the potential to reduce fuel use and carbon emissions.

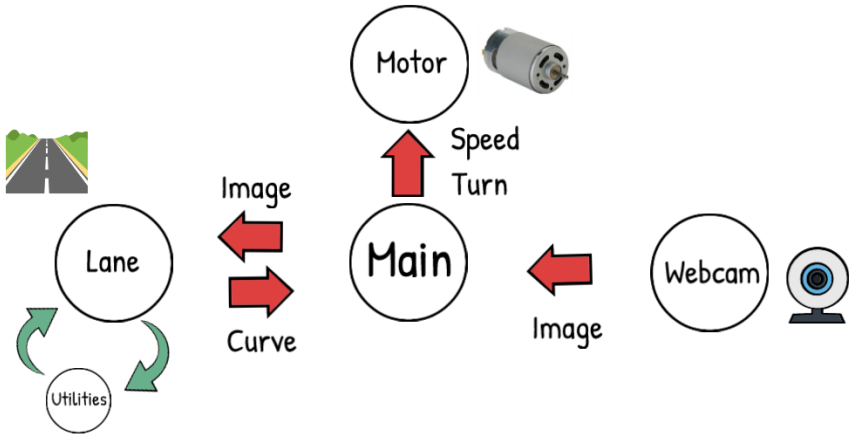
Fewer traffic jams save fuel and reduce greenhouse gases from needless idling.

PROPOSED METHODOLOGY

The Concept of Lane Detection

The idea is to find the path using colour detection or edge detector and then getting the curve using summation of pixels in the y direction i.e., a histogram. Splitting the task into 5 different steps. This includes Thresholding, Warping, Histogram, Averaging, and Displaying. Since we have been creating modules so far, we are going to create a module for lane detection as well. This way we don't need to put all the code in one script instead we can have separate python files that each perform their separate tasks. So, for this project we will have a Main Script that will be connected to our Motor Module and the Lane Detection Module. Since the Lane Detection code will take up some space, we will separate all the functions into a Utilities file keeping the main Module neat and clean.



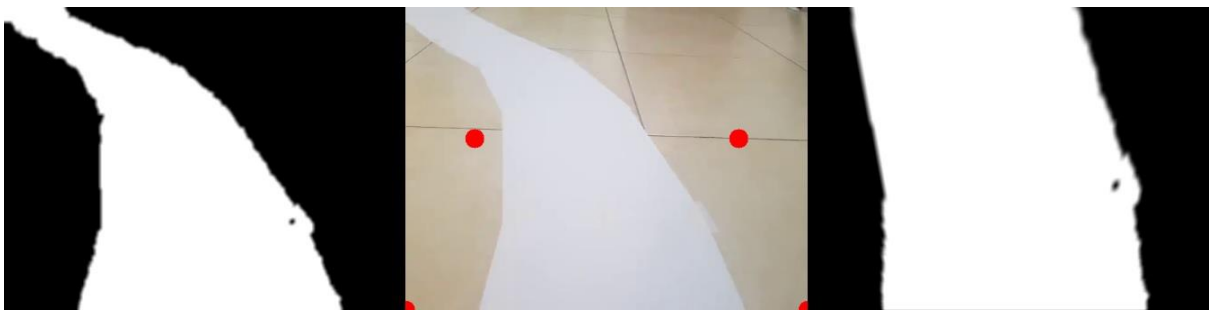


STEP 1 – Thresholding

The idea here is to get the path using either Colour or Edge Detection. Converting the image to HSV colour space and then applying a range of colour to find.

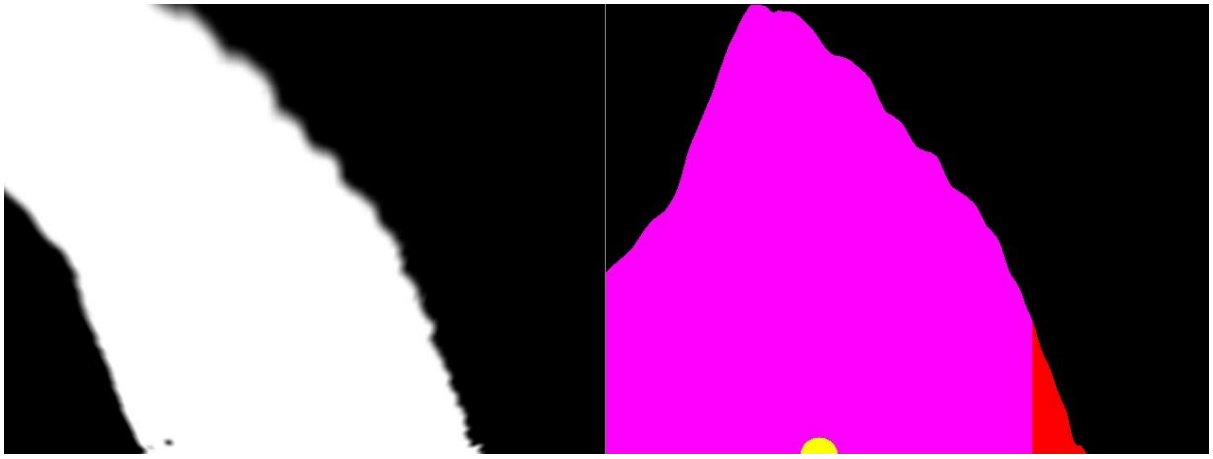
STEP 2- Wrapping

Instead of processing the whole image we just want to know the curve on the path right now and not a few seconds ahead. So, we can simply crop our image, but this is not enough since we want to look at the road as if we were watching from the top. This is known as a bird eye view and it is important because it will allow us to easily find the curve. To warp the image, we need to define the initial points. These points we can determine manually. So, to make this process easier we could use track bars to experiment with different values. The idea is to get a rectangle shape when the road is straight.



STEP 3 - Histogram

Now comes the most important part, finding the curve in our path. To do this we will use the summation of pixels. Given that our Warped image is now binary i.e. it has either black or white pixels, we can sum the pixel values in the y direction. Let's look at this in more detail.

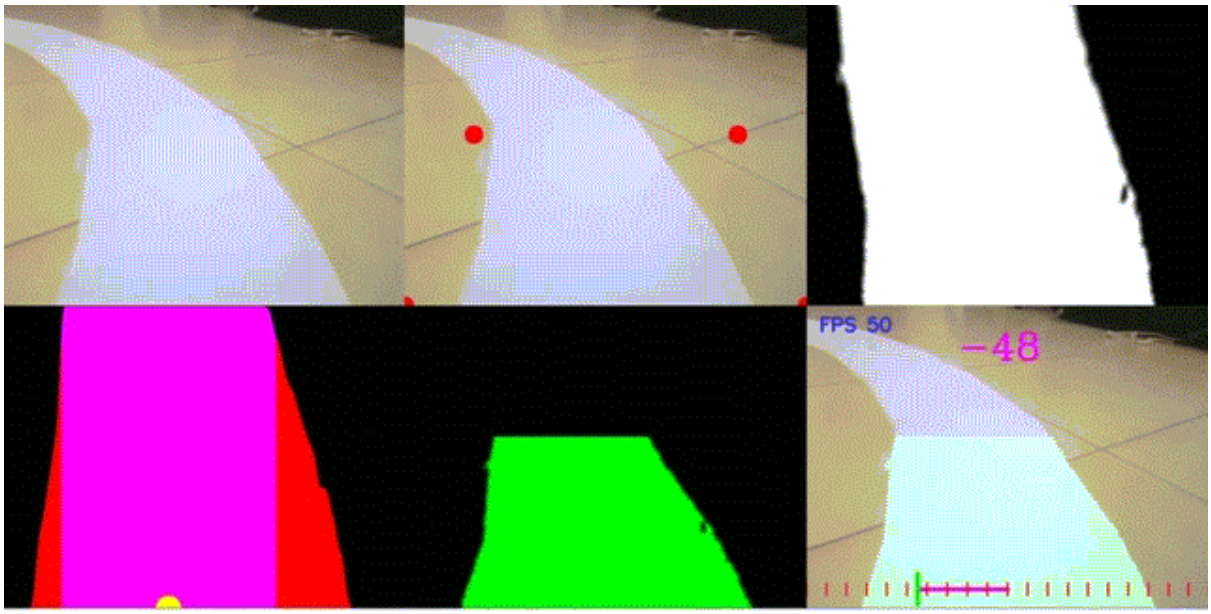


STEP 4 – Averaging

Once we have the curve value, we will append it in a list so that we can average this value. Averaging will allow smooth motion and will avoid any dramatic movements.

STEP 5 – Display

We can add options to display the final result. We will add an input argument to our main function so that we can have the flexibility of turning it on and off, since raspberry pi would run at very slow speeds if we display and run at the same time.



FACILITIES REQUIRED

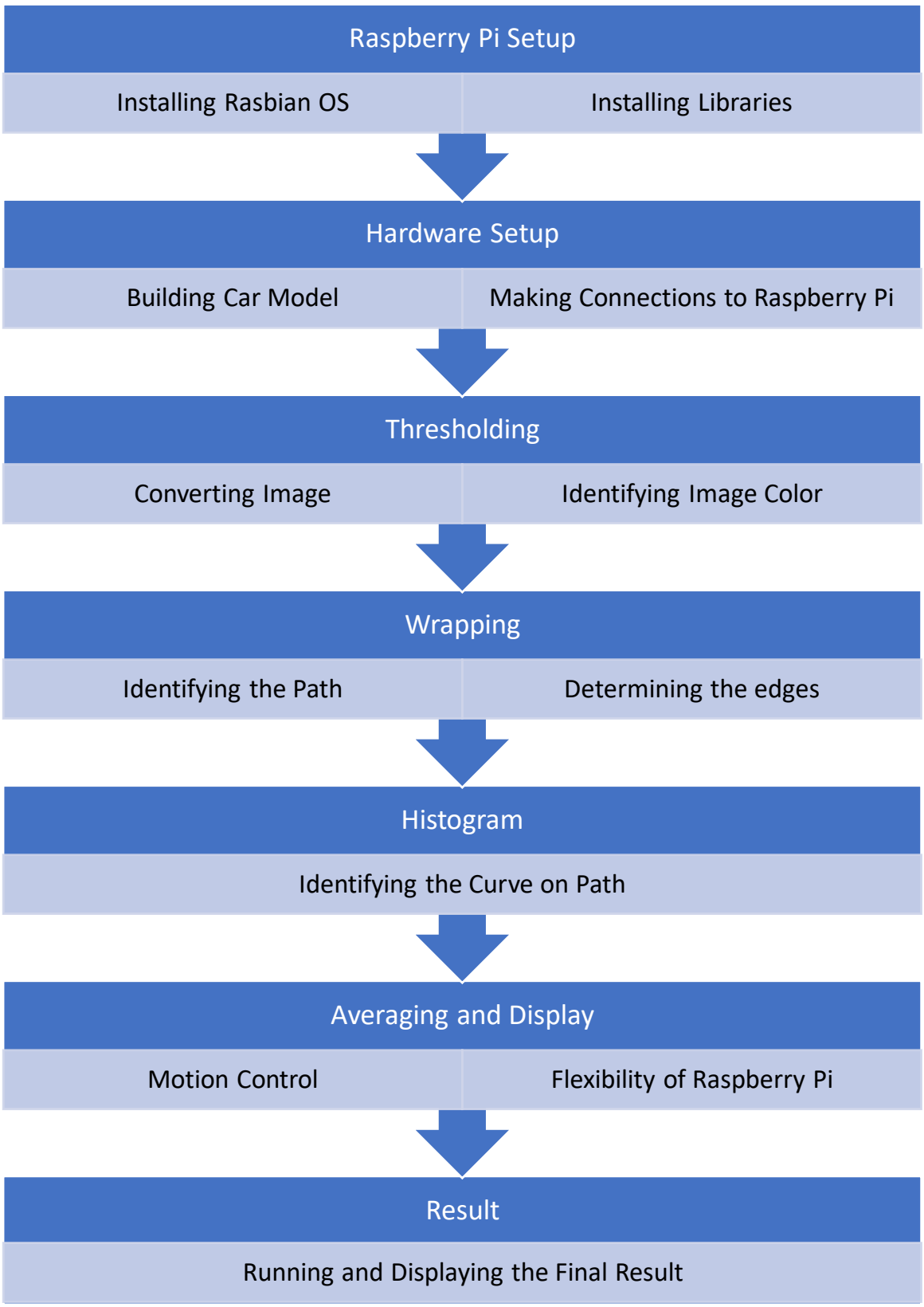
HARDWARE

1. Raspberry Pi
2. Raspberry Pi Camera Module
3. L298N Motor Driver
4. Robot Chassis
5. BO Motors
6. BO Motor Wheels

SOFTWARE

1. Python 3.7
2. Python IDLE
3. VS Code
4. Spyder Notebook
5. PyCharm
6. MU Python Code Editor for Raspberry Pi
7. Raspbian 32-bit Operating System

TIMELINE


Raspberry Pi Setup
Installing Rasbian OS
Installing Libraries
Hardware Setup
Building Car Model
Making Connections to Raspberry Pi
Thresholding
Converting Image
Identifying Image Color
Wrapping
Identifying the Path
Determining the edges
Histogram
Identifying the Curve on Path
Averaging and Display
Motion Control
Flexibility of Raspberry Pi
Result
Running and Displaying the Final Result

TEAM

1. VINAYAK LAL (2017BTCS106)	Hardware and Software Developer
2. UDIT TIWARI (2017BTCS102)	Hardware and Software Developer
3. ASHWINI JOSHI (2017BTCS026)	Quality and Assurance Analyst (Testing)
4. MEENAL SRIVASTAVA (2017BTCS050)	Quality and Assurance Analyst (Integration)

BIBLIOGRAPHY

1. https://weatherhead.case.edu/departments/design-and-innovation/student-works/deliverables/jarvis/final-project-report.pdf
2. https://coalitionforfuturemobility.com/benefits-of-self-driving-vehicles/
3. https://www.raspberrypi.org/
4. https://www.deeplearningbook.org/
5. https://www.python.org/